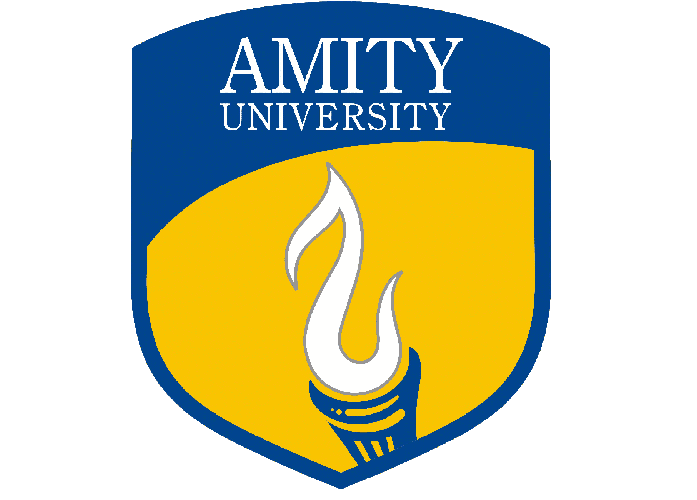
**PROJECT REPORT**

**on**

**Building Custom ALEXA using AWS**

**Submitted to:**

**Amity university, Uttar Pradesh**

****

**In partial fulfilment of the requirements for the award of the degree of**

**Bachelor of technology**

**in**

**Computer Science Engineering**

**By**

**Deeksha Bisht**

**5CSE6-Y A2305220629**

**Under the guidance of**

**Dr. Sunil Kumar**

**Department of Computer science and engineering**

**Amity school of engineering and technology**

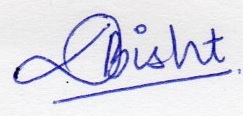
**Amity University, Uttar Pradesh**

**May-June 2022**

**ACKNOWLEDGEMENTS:**

I express my sincere gratitude to my faculty guide Dr. Sunil Kumar , for his guidance, continuous support and cooperation throughout our project, without which the present work would not have been possible. My endeavour stands incomplete without dedicating my gratitude to him; he has contributed a lot towards successful completion of my project work.

I would also like to express our gratitude to our families and friends for their unending support, and tireless effort that kept us motivated throughout the completion of this project.



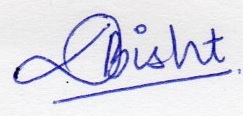
Deeksha Bisht

A2305220629

**DECLARATION:**

I Deeksha Bisht, student of BTech Computer Science Engineering hereby declare that the report titled “Building Custom Alexa using AWS” which is submitted by me to Department of Computer Science and Engineering, Amity University, Uttar Pradesh, in partial fulfilment of requirement for the award of the degree of Bachelor of Technology [ CSE ],has not been previously formed the basis for the award of any degree, diploma or other similar title or recognition. The Author attests that permission has been obtained for the use of any copy righted material appearing in the report other than brief excerpts requiring only proper acknowledged.

Text, letter

Description automatically generated 

Dr. Sunil Kumar Deeksha Bisht

Associate Professor A2305220629

Noida, Uttar Pradesh 2020-2024

Date: 20 July 2022

**CERTIFICATE:**

On the basis of declaration submitted by Deeksha Bisht, student of B.Tech (Computer Science Engineering), I hereby certify that the report entitled “Building custom Alexa using AWS” which is submitted to Department of Computer Science Engineering, Amity School of Engineering and Technology, Amity University, Noida, Uttar Pradesh in partial fulfilment of requirement for the award of the degree of Bachelor of Technology (Computer Science Engineering) is an original contribution with existing knowledge and faithful record of work carried out by him under my guidance and supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

A picture containing text, clipart

Description automatically generated

Date: 10/07/2022

Dr. Sunil Kumar

Associate Professor

Department of Computer Science and Engineering

Amity University, Noida, Uttar Pradesh

**TABLE OF CONTENTS:**

|  |  |  |
| --- | --- | --- |
| Serial number | Contents | Page number |
| 1. | Abstract | 6 |
| 2. | Introduction-Cloud computing and Evolution of AWS | 7 |
| 3. | Rise of AWS and its merits | 9 |
| 4. | Alexa Skill Set and Voice UIs | 11 |
| 5. | Related Work and Proposed Idea | 13 |
| 6. | AWS Lambda | 14 |
| 7. | Materials and Methodology | 15 |
| 8. | Result and Discussion | 25 |
| 9. | Conclusion and Future Research | 26 |
| 10. | References | 27 |

**ABSTRACT**

Cloud computing is the delivery of on demand services that includes servers to transfer from physical infrastructure, database to store glob of data over the internet. You can implement almost anything on your application.

AWS or Amazon Web Services is a software for cloud computing. This service provides many fully featured services to compute, store, secure or transfer a whole offline enterprise to cloud. One of the biggest features being flexible to any technology, cost effective, scalable and pay-as-you-go services.

Alexa Skills Kit (ASK) is a software supported by AWS for the development framework. It helps to build content or skills for software like Alexa. Alexa is an interactive voice-control interface which performs simple activities like telling the time or playing a certain song. Now-a-days many devices have in built Alexa, for example Amazon Echo, Amazon Fire TV and more.

Voice user interfaces are advantageous as it promotes hands-free, settling way to use the application while still doing our prime work. Increased pace, pre-requisite knowledge of the system, multi-tasking and accessibility are more merits of this software.

AWS services will be like AWS Lambda, EC2 and many more compute and scaling service will be used to achieve the main objective of this project. This custom-made voice UI will run on the code provided will the service does the rest of the job. Although trying my best to create the perfect prototype for this custom Alexa, there is always scope of improvement towards perfecting any task.

**INTRODUCTION**

Cloud Computing- Future of fintech and all type of business industry, a career opportunity and an ocean of storage. Cloud Computing is a platform of delivering computing services like storage, database, networking, intelligence and much more. These services are retrieved or uploaded on the ‘cloud’. Cloud is an online vast storage of servers which loads and unloads these services as per client request. Cloud computing serves many benefits like:

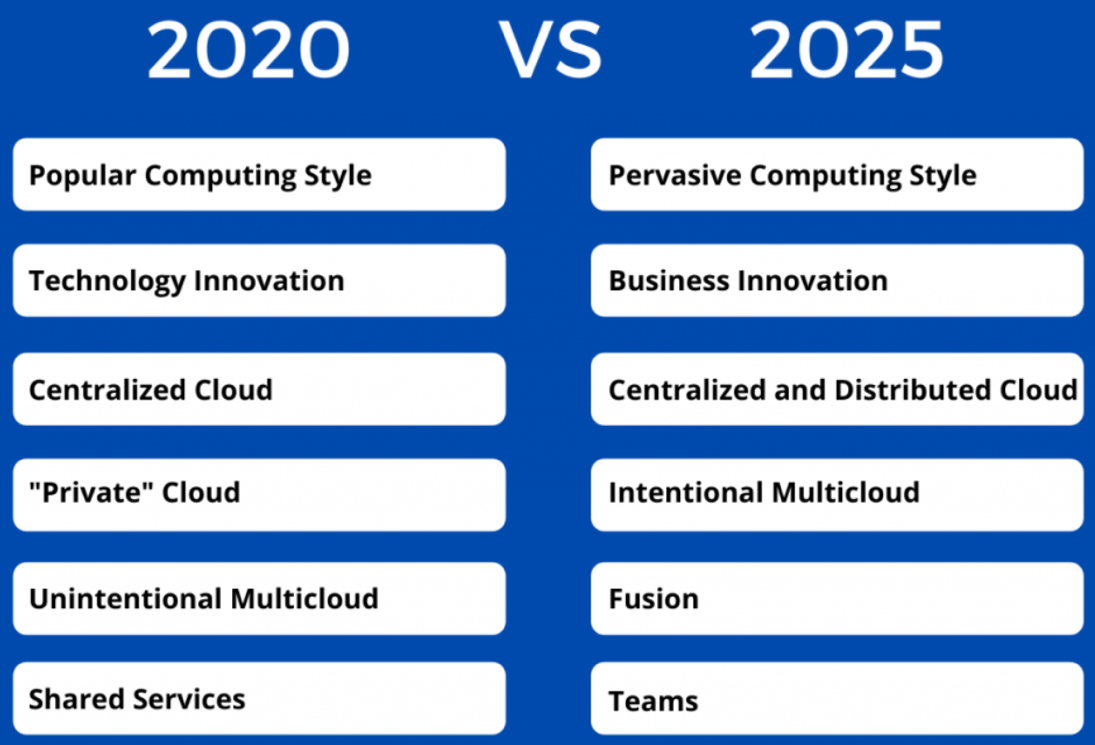
(AWS Academy, n.d.)

1. Trade capital expenses for variable expense – Variable expense is a cost that can be avoided anytime. Investing in heavy data centres, assets, properties when not sure if all of them will be used or not is something that can be avoided. Cloud provides services at reasonable cost with not taking any physical storage.
2. Mass economy of scale - Using cloud computing, a lower variable cost can be achieved than buying own physical assets. Usage of services by hundreds of thousands of customers is aggregated in the cloud, service providers such as AWS can achieve higher economy of scale, which turns into lower pay-as-you-go prices.
3. Un-guessable storage capacity – You can use as much as services or storage you need and can even scale it up or down.
4. High speed – Resource availability is no problem while using these services.
5. Huge decrease in capital expense – Focusing on skills you need reduces the other expenses exponentially.
6. Global connectivity in minutes – With multiple AWS regions all around the world, you can deploy your applications easily.

Cloud computing is evolving as we talk about it. The evolution is divided in three phases:

1. Idea – peaked in 1960s to develop grid computing that lasted till internet bubble era.
2. Pre-Cloud phase – resurfaced in 1999 and was in talk till 2006. This phase talks about AaaS i.e. Application as a Service.
3. Cloud Phase – In 2007, cloud developed and got differentiated into the three forms as known today- IaaS, PaaS and SaaS

The anticipated change from 2020-2025 is declared as:

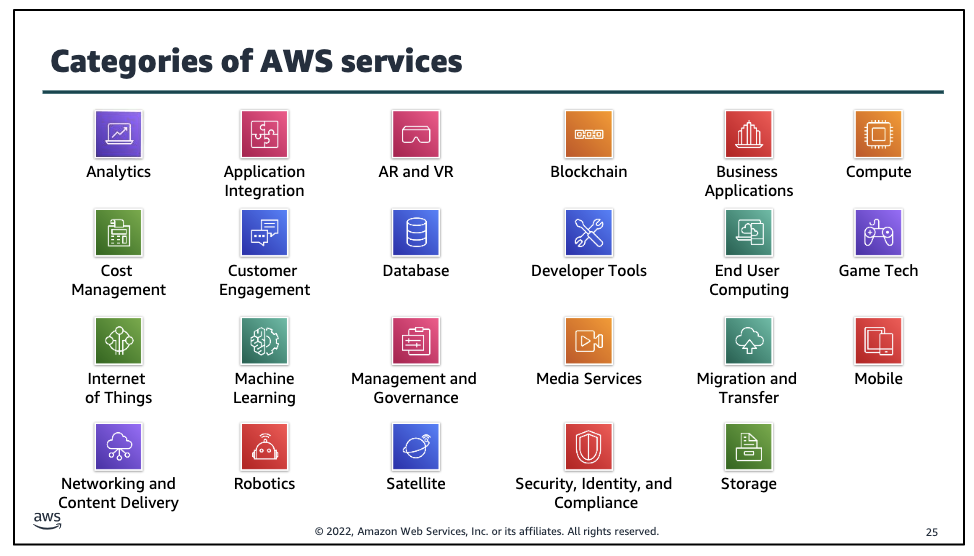


Cloud Computing has increased its scope to India too. With a whooping cost of $2 billion and still with expectation to grow with annual rate of 30%. There are many roles associated with this area such as:

* Cloud Infrastructure Engineer
* Cloud Architect
* Cloud Software Engineer
* Cloud Enterprise Architect

**RISE OF AWS and MERITS**

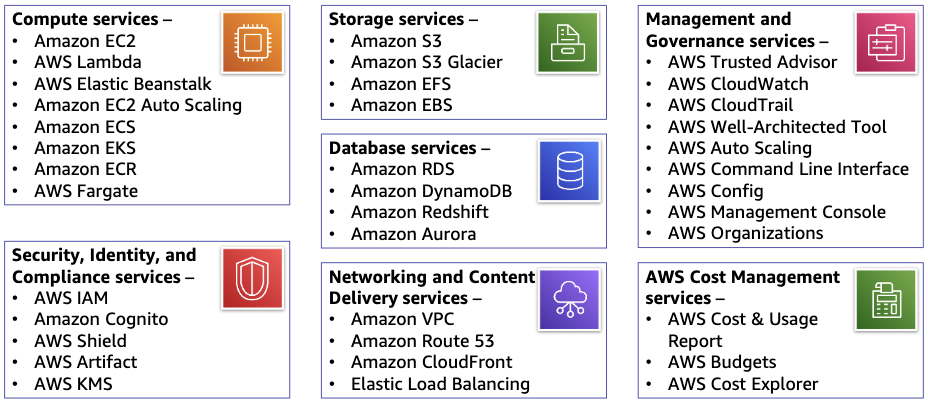
AWS or Amazon Web Services is the world’s most adopted cloud platform offering over 180+ full services across the global. A web service provided by Amazon which is easily available over the cloud. With a standardized format, platform independent programming language, this service is now soon to be declared as the future of cloud computing. Services provided by AWS are categorised as shown in the picture below:



This platform claims itself as one of the leading for the following reasons:

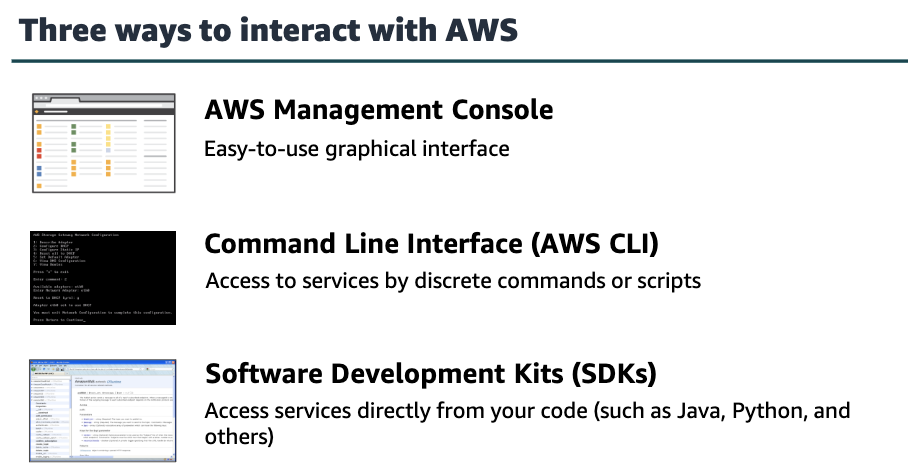
1. Numerous functionality: AWS provides the most amount of services as compared to any other cloud platform. Covering services from infrastructure management to computing, storage and autoscaling, this emerging technology makes your applications faster, easier to handle and more cost effective.
2. Flexible: Having millions of start-ups, enterprises and public sector organisation using every single case on AWS. Having so many different functionalities makes AWS easier to adapt to.
3. Security: the core engine of AWS satisfy security for military, global banks and other highly sensitive organisations.
4. Innovation: As discussed earlier, with high flexibility and adaptability in services, AWS is accelerating its pace of innovation to a whole new level.

The services provided by AWS are:



To access these services there are three ways:

1. AWS Management Console – a GUI for majority of features.
2. AWS Command Line Interface – utilities can be launched using Linux, MacOS, Windows
3. Software Development kits – packages that can access numerous programming language.



**ALEXA SKILL SET and VOICE UIs:**

With the emerging set of AWS services, Voice-user interfaces technology are the new talk in the market. A hassle-free way to fulfil your commands without getting distracted from your prime work.

These skills are accessed using certain keyword. Here, Alexa in the device uses the phrase: “Alexa”. The device streams the speech to the Alexa service in the cloud. The skill runs as a service on the cloud platform.

1. Pace- (htt1) According to a Standford study, it is found out that verbal communication is four times faster than written communication. This makes the voice-user interface better to use for tasks like searching through a large record log.
2. Knows the system/hardware- For example calling Siri to find out the balance of audio/video on an iPhone. Siri knows the whole system inside out so it is easily fetched by it.
3. Multi-tasking- Voice interface can start and tackle multiple tasks at one.

The table given below describes above points:

|  |  |  |
| --- | --- | --- |
| Action | Using Voice user Interface | Using typical way |
| Check today’s whether | Say “Alexa, what is the weather like today?” | Unlock your device < go to weather app/ search through the internet |
| Play a song | Say “Alexa play ‘song name’ (can give extra information of which app to use)” | Unlock your device < go to Music app < search the song |

METHODOLGY:

This project reflects on how to build a custom Alexa using the services provided by AWS. A simple algo first written then converted to program will be entered into an API or command-line interface.

Voice UIs:

Saying our thoughts instead of writing them is an ease and convivence for human kind. Voice UIs are voices built-inside phone, laptops and other electronic device which acts as our assistant and do our jobs for us. For example, if someone wants to play a movie on TV, simply ask the voice control to play it for you. From senior people to young toddlers, everybody has become addicted to just saying and leaving it to the voice UI to do their work. This piece of technology is used almost every day and by everybody. Therefore innovation like availability in multiple language is a new feature added to this.



**RELATED WORK**

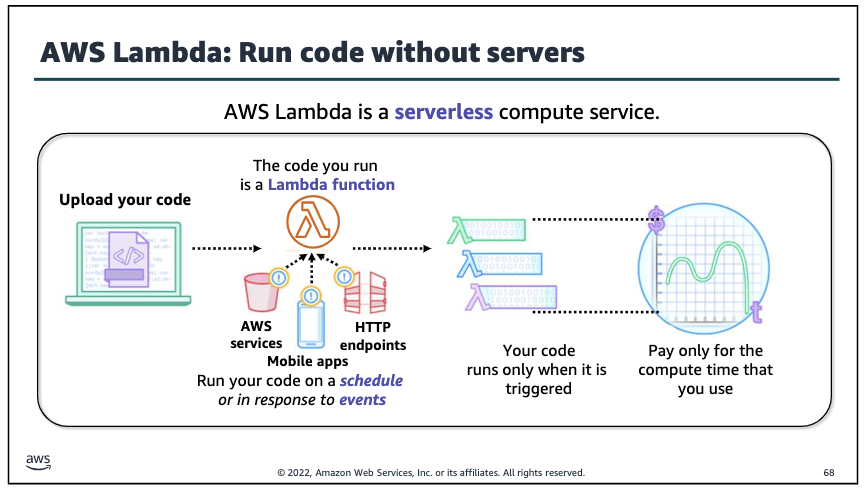
* Privacy - Although these assistant are not that highly AI software, but as they are innovating user privacy is at risk. Engineers are taking steps to ensure security towards a user privacy.
* Multiple voices – Now a days the built-in assistant have different accents and language like Spanish, Hindi, Portuguese for maximum worldwide usage of this technology.
* These assistant are also helping illiterate people by giving them information in laymen language.

**PROPOSED IDEA**

This project focuses on developing a simple voice assistant to just understand the working and optimization of software and AWS as a profession. This report will be a conclusion as a whole about how to create Voice UIs and what all command can be coded using AWS functions.

**AWS LAMBDA**

An Amazon Web Service which offers to run your code without any servers. An event driven function which runs and scales automatically when needed, so there is no need to provision or continuously run servers.



AWS Lambda supports code written in several languages, including Node.js, Java, and Python. For some languages, you can edit the code in the inline code editor in the AWS Lambda console. For basic testing, you can invoke the function manually by sending JSON requests in the Lambda console. Lambda functions generally have to be triggered. It can be either on time slots or reaction to certain events.

MERITS OF USING LAMBDA:

1. Support multi language programming
2. No manual administration needed
3. Pay as you use pricing
4. Built-in fault correction

**MATERIALS and METHODS**

MATERIALS:

In this project, building a Custom Alexa requires AWS services as the prime material. The Algorithm behind the task presented upfront is the other prime factor required.

Links to use for developing Alexa Skill Set:

https://developer.amazon.com/en-GB/alexa/alexa-skills-kit

METHOD:

Host a Custom Skill as an AWS Lambda Function

The easiest way to build a cloud-based service for custom Alexa skill is to use AWS Lambda, After you upload the code for the Alexa skill to a Lambda function, Lambda does the rest, executing it in response to Alexa voice interactions and automatically managing the compute resources for you.

## About Lambda functions and custom skills

Lambda function eliminates complexity of setting up and managing your own endpoint:

* You do not need to manage or manage any computing resources for your service.
* You do not need an SSL certificate.
* Manual verification for transmitting of requests is not needed. Access to run your function is instead controlled by permissions within AWS.
* AWS Lambda runs your code only when you need it and adapts to your usage, eliminating the need to provision or run servers 24/7.
* Alexa encrypts its communication with Lambda using TLS.
* For most developers, the free tier of Lambda is sufficient for the feature supporting the Alexa skill. Monthly first million requests are free.

The Lambda function must process the requests sent to the skill by Alexa. Lambda functions for custom skills can be hosted in any AWS Lambda region. Select an optimal region for the function to determine the optimal choice for your skill configuration.

## Select the optimal region for your AWS Lambda function

You are cued to select a region while creating an AWS Lambda function. It is generally advised to use the commonly recommended region. This choice means you reduce latency between Alexa and your skill service. However, if your skill is using other resources that are hosted in a different region, such as DynamoDB databases, Amazon S3 content, or any external services, it might be best to select the region that is closest to those resources in order to reduce overall latency for your skill, even if that region is different from the recommendation in this table. Below are the generally provided regions in AWS servers:

| **ASK region code** | **Commonly recommended AWS region code** | **Commonly recommended AWS region name** |
| --- | --- | --- |
| NA | us-east-1 | US East (N. Virginia) |
| EU or IN | eu-west-1 | EU (Ireland) |
| FE | us-west-2 | US West (Oregon) |

| **Allowed AWS Lambda regions** | | |
| --- | --- | --- |
| US East (Ohio) | EU (Frankfurt) | Asia Pacific (Hong Kong) |
| US East (N. Virginia) | EU (Ireland) | Asia Pacific (Mumbai) |
| US West (N. California) | EU (London) | Asia Pacific (Seoul) |
| US West (Oregon) | EU (Paris) | Asia Pacific (Singapore) |
| Canada (Central) | EU (Stockholm) | Asia Pacific (Sydney) |
| South America (Sao Paulo) | Middle East (Bahrain) | Asia Pacific (Tokyo) |

## Use a template to create your Lambda function

While using the Alexa Skills Kit SDK for Node or Python, you can use a template provided in the AWS Serverless App Repository to create your function. This automatically creates the resources you need for the function to work with Alexa:

* The Lambda function itself, with starting sample code from the corresponding GitHub repository. You can use this to experiment with the sample functionality, or replace it with your own code.
* The SDK dependencies needed for the code. This eliminates the need to upload the dependencies (such as node\_modules) yourself.
* The trigger that grants Alexa the necessary invocation permissions for your function.
* The **role** for the function. This defines the AWS resources the function can access.

This serverless app repository lets you find applications to deploy. Deployment of an application is necessary to create the resources associated with the application in an AWS CloudFormation stack. Once deployed, the Lambda function normally.

### Direct links to recommended Alexa samples

Click the link

* **Node.js** — [alexa-skills-kit-nodejs-premium-facts-skill](https://console.aws.amazon.com/lambda/home?region=us-east-1" \l "/create/app?applicationId=arn:aws:serverlessrepo:us-east-1:173334852312:applications/alexa-skills-kit-nodejs-premium-facts-skill)
* **Python**—[alexa-skills-kit-python36-factskill](https://console.aws.amazon.com/lambda/home?#/create/app?applicationId=arn:aws:serverlessrepo:us-east-1:173334852312:applications/alexa-skills-kit-python36-factskill)

### Search for an Alexa sample

Alternatively, you can search for Alexa samples in the serverless app repository:

1. Go to AWS and login or create an account.
2. Login to the AWS Management Console and navigate to AWS Lambda.
3. Click **Create function** >> **Browse serverless app repository**.
4. Under **Public applications**, type Alexa in the search box.
5. Click the name of the desired application. The application details page will appear his provides you with a detail page about the app. Details about the resources that will be displayed and created under “Template”

### Deploy the application to create your function

1. Open the page for the serverless application to deploy.
2. From the **Region** drop-down list in the upper-right corner of the console, select one of the allowed AWS regions.
3. In the **Application Settings**section, optionally modify the following:
   * Application name
   * SkillDescription
   * SkillFunctionName

Note : Deployment with the same template multiple times is possible but it is advised to use a unique name application each time.

1. Click the **Deploy** button at the bottom of the page.
2. Wait for the status of all resources to change to **CREATE COMPLETE**
3. Under **Resources**, click the link to open the newly created function in the Lambda console.

## Create a Lambda function from the start

This method is used if ASK SDK for Node v2 or ASK SDK for Python are not being used.

**To create a new Lambda function**

1. Go to AWS and login or create an account.
2. Login to the AWS Management Console and navigate to AWS Lambda.
3. Select one of the allowed AWS regions from Region drop-down.
4. Click **Create a Lambda Function**.
5. Assign a **Name**and the role of the function. This defines the AWS resources the function can access To use an existing role, select the role under **Existing role.**
6. Select language as you desire to use for the **Runtime.** While using one of the Alexa Skills Kit SDKs, select Node, Python, or Java.
7. Click **Create function.**

### Define a new role for the function

Role specifies the AWS resources what your function can access. To create a new role while configuring your function:

## Lambda Function Handler and role < Role , create a new role template(s).

## Enter Role name.

## Policy Template < select Simple Microservice permissions.

To view or edit the code, click the function name under **Designer**, then scroll down to the **Function code** section.

Connect the Lambda function to your skill

Update the endpoint field of the skill to AWS Lambda ARN for the skill to connect the function with your skill.

1. Copy the **ARN** shown in the upper-right corner of your Lambda function.
2. Open or create your skill in the developer console.
3. **Custom > Endpoint**.
4. For the **Service Endpoint Type**, select **AWS Lambda ARN** and paste the ARN for your Lambda function in the **Default Region** box.

## Configure the trigger for a Lambda function

You must configure at least one triggerfor your function to grant Alexa the necessary invocation permissions for your function.

While adding the Alexa Skills Kit as a trigger, Amazon recommends that you also enable skill ID verification for the function and provide the skill ID*(*also called the application ID*)* for your skill. This aids that your function can be invoked only if the skill ID in the Alexa Skills Kit request matches the skill ID configured in the trigger.

### Add an Alexa Skills Kit trigger

Before you add the trigger, copy the skill ID from the developer console:

1. Open or create your skill in the developer console.
2. Find the skill in the list and click **View Skill ID** below the skill name. Copy the ID from the popup.

Once you have your skill ID, add the trigger to the function:

1. AWS Management Console < AWS Lambda.
2. Click your function in the list to open the configuration details.
3. Configuration < **Designer** section < under **Add triggers**, click **Alexa Skills Kit** to select the trigger.
4. Under **Configure triggers**, select **Enable** for **Skill ID verification**.
5. Enter your skill ID in the **Skill ID**edit box.
6. Click **Add**.
7. Click **Save** to save the change.

It is highly recommended to put restriction to just Alexa and enable skill ID verification to protect your function from malicious callers.

### NOTE: You can furthermore remove an Alexa Skills Kit trigger or change an existing trigger to optimize your skill using an API or Command Line Interface.

### Configure triggers with the AWS CLI or Lambda API

AWS Lambda is compatible with both an API and Command Line Interface. It can manage Lambda functions. These consoles add a resource-based policy granting the ASK permission to invoke the function.

* For the API, use AddPermission.
* For the AWS CLI, use the add-permission command.

For the ASK trigger, you must set the following:

* action must be lambda:InvokeFunction.
* principal must be alexa-appkit.amazon.com.
* event-source-token is the ID for your skill.

For example, this CLI command adds the trigger to the Lambda function

test\_1:

aws lambda add-permission

--function-name test\_1

--statement-id 1

--action lambda:InvokeFunction

--principal alexa-appkit.amazon.com

--event-source-token amzn1.ask.skill.xxxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx

Triggers added with the API or CLI are displayed in the Lambda console just like all other triggers.

## Test a Lambda function in the console

Lambda function can be manually tested in the Lambda console by sending sample JSON events formatted in the same way as requests sent by Alexa. Sample events are required to test within the console. Function triggers are not checked, so the test event does not need a matching skill ID when using skill ID verification.

To test your function, create a new test event:

1. Function list < click the function name to open details of the function.
2. In the upper right corner of the screen, click the **Select a test event**drop-down list and click **Configure test events.**
3. Select **Create new test event**and then select one of the sample Alexa requests from the **Event template**list:
   * Alexa Start Session
   * Alexa Intent (multiple samples available)
   * Alexa End Session
4. Enter an **Event name**.
5. Click **Create**.

On the main function configuration page, make sure your event is selected and click the Test button.

After the function runs, the Execution result section displays the response by the function, in JSON format. You should see a response appropriate for the request pasted into the Sample event box. For example, for a LaunchRequest, a function built from the alexa-skills-kit-nodejs-premium-facts-skill template returns a response similar to this:

{

"version": "1.0",

"response": {

"outputSpeech": {

"type": "SSML",

"ssml": "<speak>Welcome to Testing1. To hear a random fact you can say 'Tell me a fact', or to hear about the premium categories for purchase, say 'What can I buy'. For help, say , ‘Blue'... So, What can I help you with?</speak>"

},

"reprompt": {

"outputSpeech": {

"type": "SSML",

"ssml": "<speak>I didn't catch that. What can I help you with?</speak>"

}

},

"shouldEndSession": false

},

"userAgent": "ask-node/2.3.0 Node/v8.10.0",

"sessionAttributes": {}

}

The Log output section shows any log messages generated by the code. The sample writes a log message for each type of request as follows:

2015-05-18T23:53:22.357Z 0f885f98-fdb9-11e4-80af-1b9f8363b496 onIntent requestId=amzn1.echo-api.request.6919844a-733e-4e89-893a-fdcb77e2ef0d, sessionId=amzn1.echo-api.session.abeee1a7-aee0-41e6-8192-e6faaed9f5ef

COMPARISON WITH EXISTING TECHNOLOGIES:

|  |  |
| --- | --- |
| This project  (Custom Alexa – BLUE) | Existing Technology  (Alexa, Google, Siri) |
| This project used only AWS services to develop the prototype | Voice UIs have multiple languages and platform which supports at the back end. |
| This prototype has scope for feasibility | Already a feasible technology. |
| Does simple task like playing music, Telling the weather, calculating | Does tasks from playing music to booking flights. |
| This prototype has not attained the capacity to be uploaded on a hardware | Are built-in in devices |
| Simple code | High complexity compared to Blue |

**RESULT and DISCUSSION**

This experiment was successfully conducted. This prototype can be produced and optimized further and can reach its peak. This version has attained to do simple task. Learning AWS features and preparing to take it as a career is also considered.

In this project, the custom Alexa skill i.e. Blue is able to perform tasks like playing music on desired apps, telling weather, calling someone, calculating numbers etc.

Although the prototype is at its initialise phase, I believe including more assets and more goal oriented focus will surely guarantee success.

**CONCLUSION and FUTURE RESEARCH**

With so many services now-a-days, it has become a hassle-free technique to create an Alexa setup and use it commercially.

Future purposes are:

* 1. Maximum optimization
  2. Monetary ads can be uploaded with the assistant so when the user connects to internet, he gets a set of customised ads.
  3. Increased security using voice recognition with addition to thumbprint or facial recognition for securing the hardware.
  4. Using voice assistant as guides in tourist areas.

**REFERENCES**

<https://awsacademy.instructure.com/courses/13414/modules/items/1147003>

<https://developer.amazon.com/en-US/docs/alexa/custom-skills/host-a-custom-skill-as-an-aws-lambda-function.html#define-new-role>

https://developer.amazon.com/en-US/docs/alexa/ask-overviews/what-is-the-alexa-skills-kit.html

GitHub Link for the code:

---------------------------------------------------

https://github.com/BlueDesigning/BLUE\_CODE.git